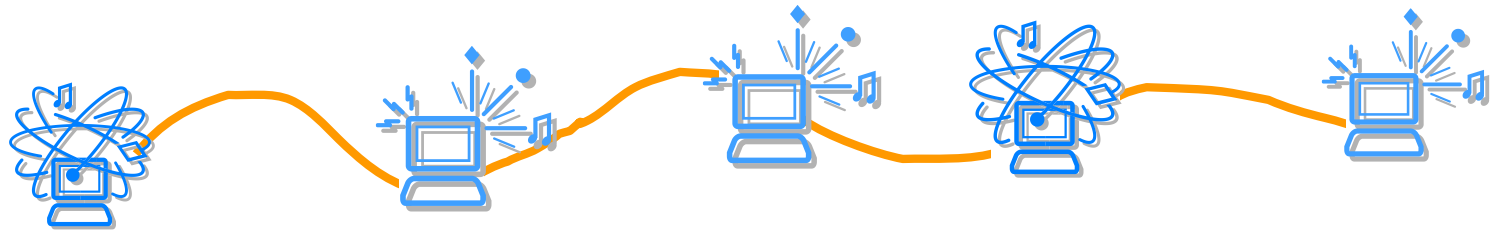
Protocolos de Encaminhamento



Construção das tabelas

Protocolos teóricos

Routing

- Origem da informação
- Estrutura das tabelas
- Base histórica
- Propagação automática de caminhos
- Algoritmos *Vector Distance* (VD) e *Link State* (LS)

Routing

- Problemas do encaminhamento
  - Que valores colocar nas tabelas de encaminhamento?
  - Como obter esses valores?
  - Como saber se esses valores indicam os caminhos mais curtos?
  - Depende da complexidade da arquitectura e das políticas de administração da Internet

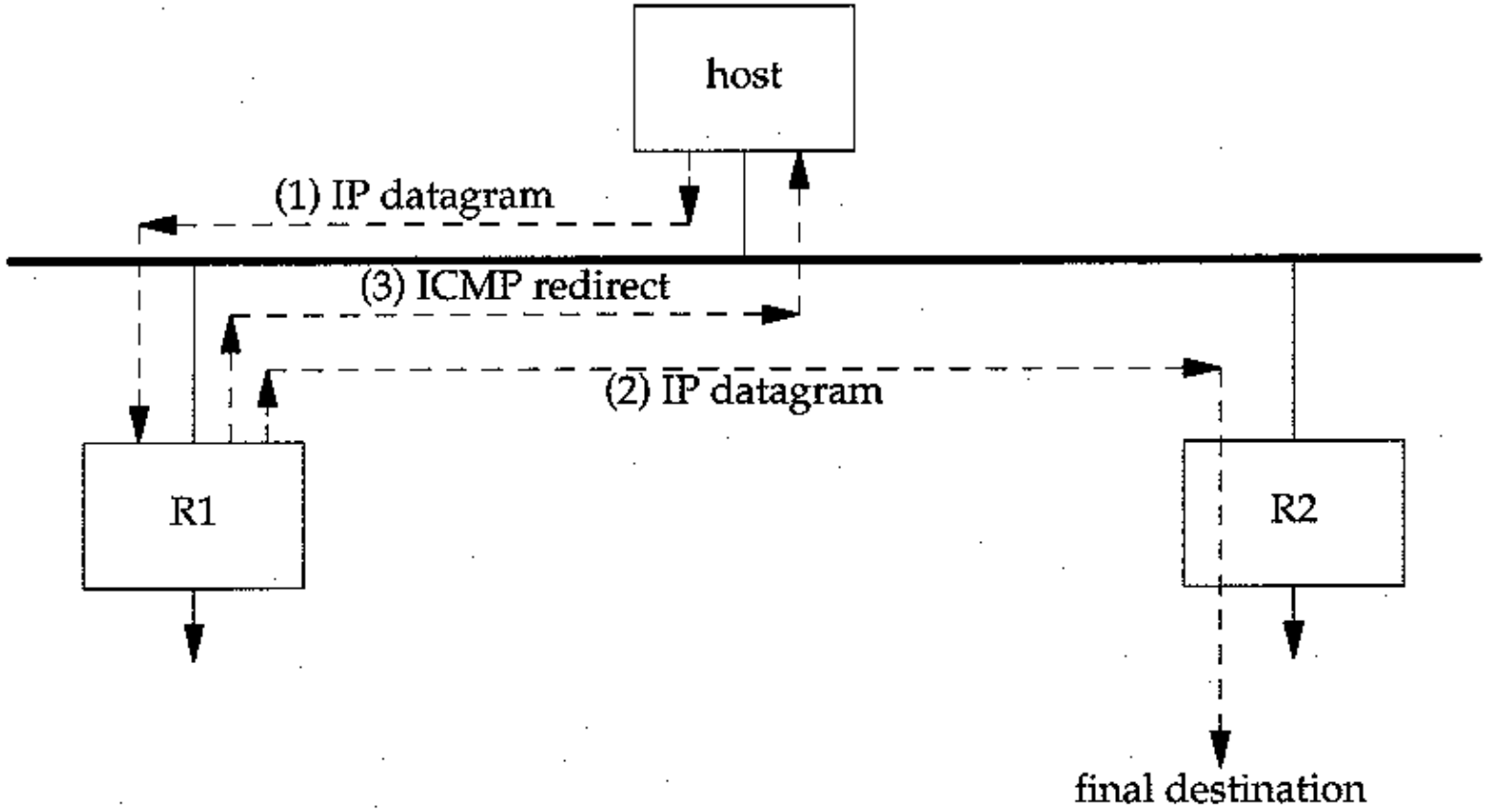
Encaminhamento vs envio (*Routing* vs. *Forwarding*)

- Envio (*Forwarding*):
  - Seleciona uma porta de saída baseada no endereço destino e na tabela de encaminhamento

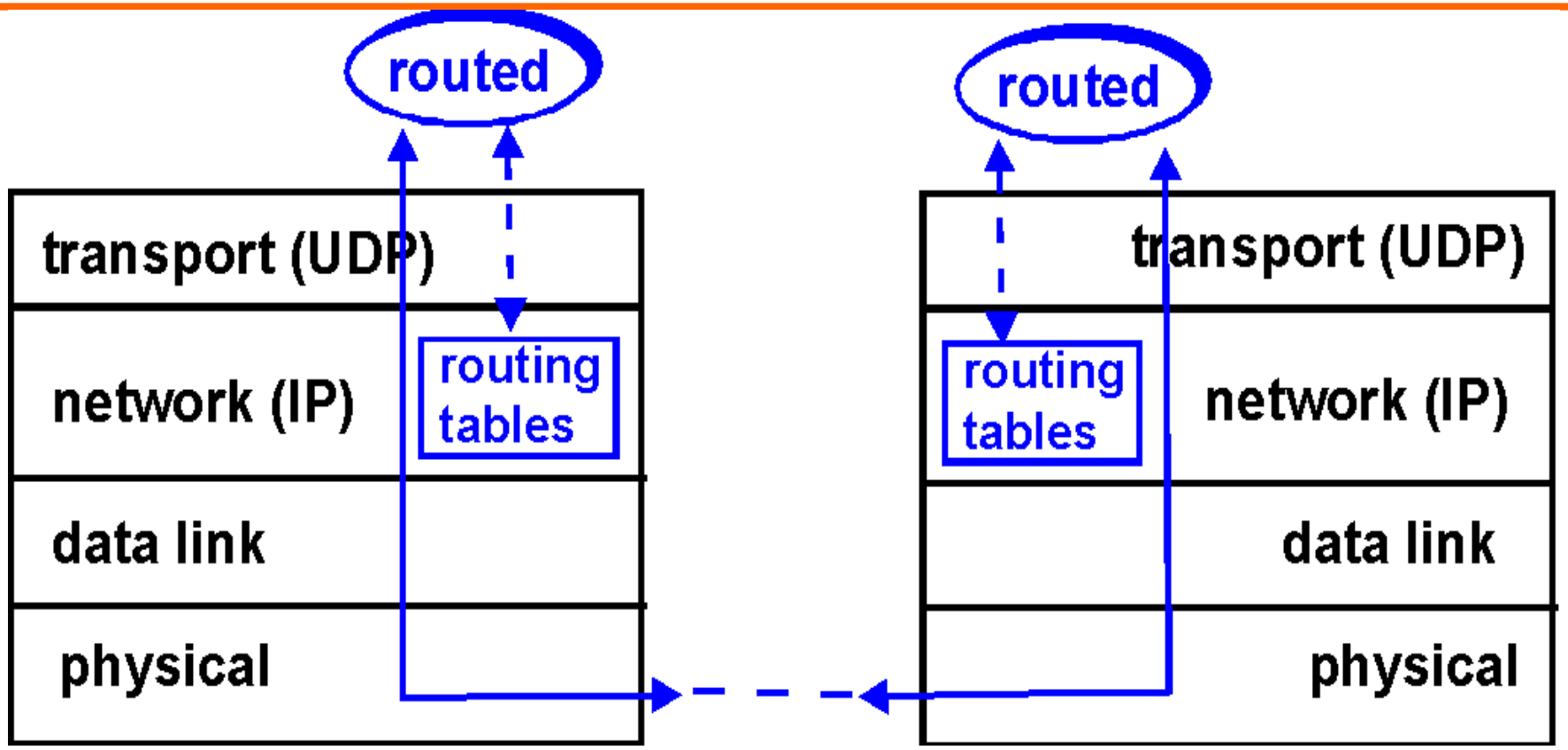
- Encaminhamento (*Routing*):
  - Processo pelo qual a tabela de encaminhamento é construída
  - Função de encontrar os caminhos numa rede.

Mensagem de "*Redirect*" do ICMP

- *Redirect*
  - O *router* informa que o datagrama devia ter sido enviado para outro *router*

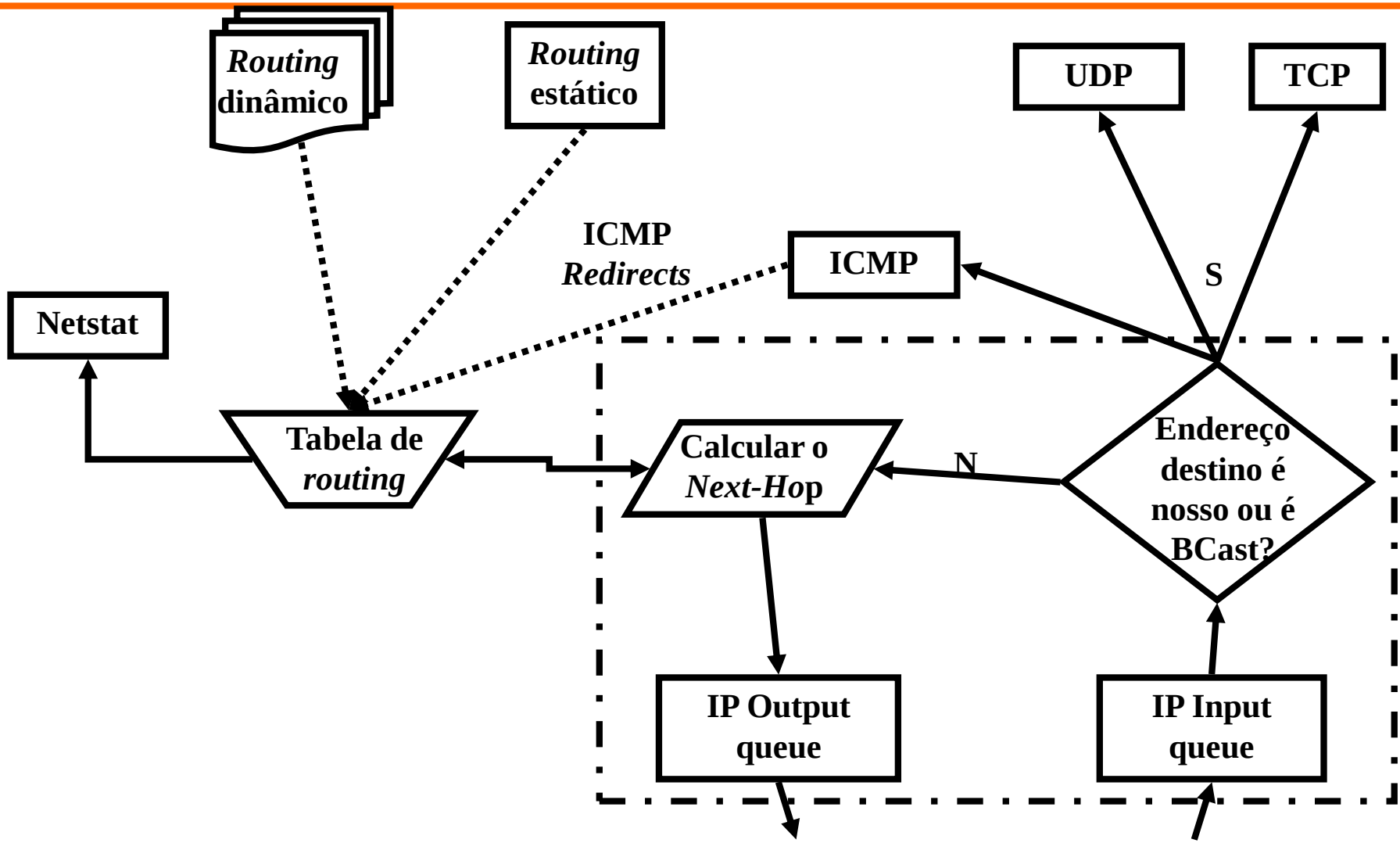


Modelo do Routing Dinâmico

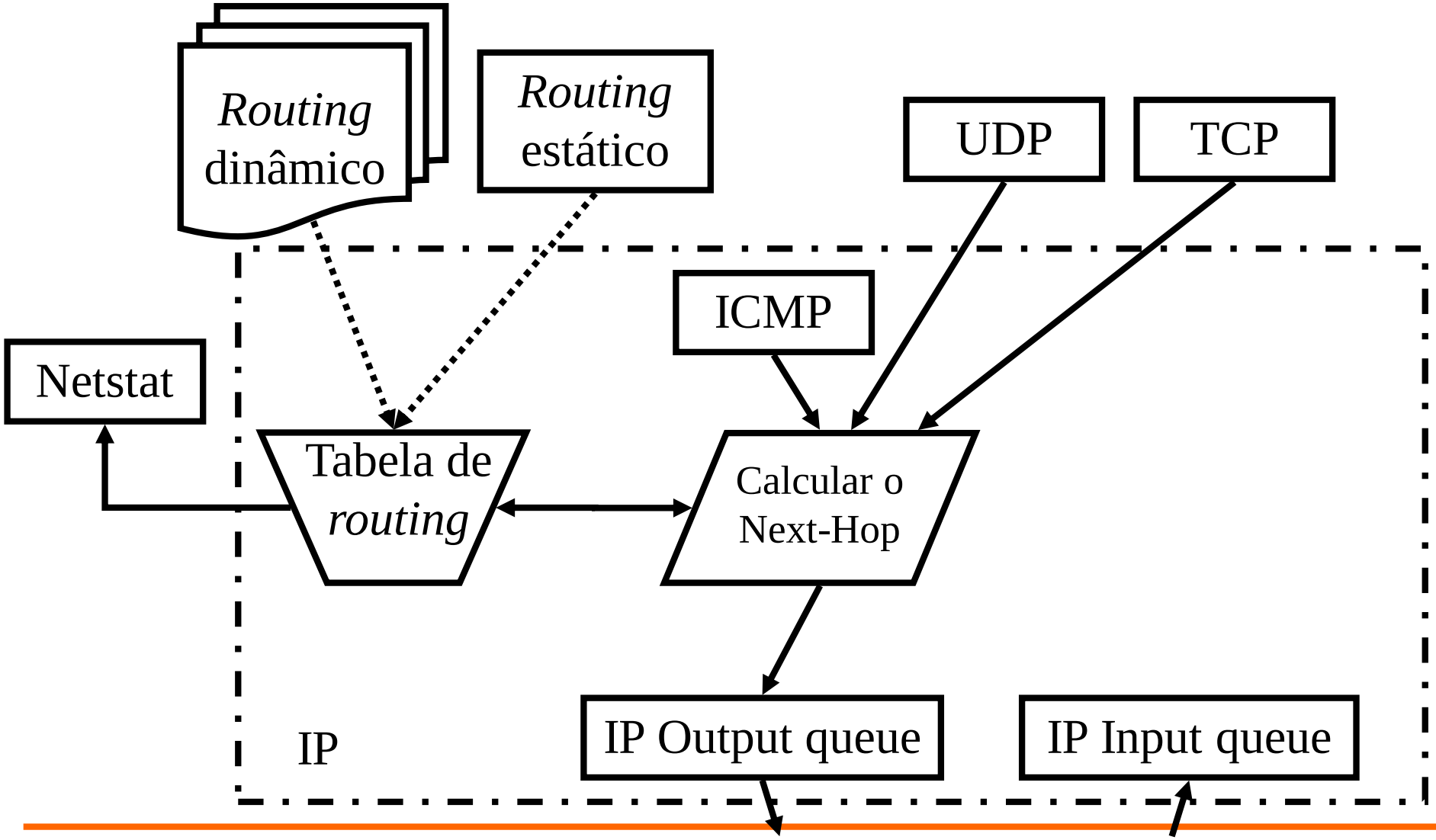


- Um *router* toma uma decisão localmente acerca da topologia global

Routing IP: pacotes recebidos



Routing IP: pacotes gerados



Routing com informação parcial

- Contexto
  - Tipicamente uma máquina ou *router* não tem informação sobre todos os destinos possíveis

- Quem pode encaminhar com informação parcial ?
  - As máquinas (*hosts*)
    - Têm configurado um *router* por omissão (*Default Router* ou *Gateway*) em quem confiam para fazer o encaminhamento
  - Os *routers*
    - Conhecem rotas para alguns destinos
    - Podem ter caminhos por omissão
      - Que, desde que as tabelas de *routing* globais estejam consistentes, garantem que é possível chegar a qualquer destino

Algoritmo de *routing*

```
struct EntradaTR {
    EndIP    endRede;     // Endereço de rede destino
    EndIP    mascara;     // Mascara de rede
    EndIP    proxRouter;  // Endereço do próximo router a enviar
    EndIP    interface;   // Endereço da interface por onde enviar
};

int EncaminhamentoDatagramas (Datagrama d, TabelaRouting tr)
{
  EntradaTR etr;
  EndIp dst = ExtrairEndereçoIP(d);
  while ( (etr = PróximaEntradaTR(tr)) != null)
      if(PertenceRede(dst, etr.endRede, etr.mascara)) {
            if(PossuiInterface(etr.proxRouter))
               EnviarDatagrama(d, dst, etr.interface);    // Entrega directa
            else
               EnviarDatagrama(d, etr.proxRouter, etr.interface);    // Entrega indirecta
            return OK;
      }
    return NO_ROUTE;
}
```

Tabela de *routing* de uma máquina UNIX

- Machina> **netstat -rn**
- Kernel IP routing table

Destination	Gateway	Genmask	Flags	MSS	Window	irtt	Iface
62.48.131.0	172.25.52.252	255.255.255.224	UG	40	0	0	eth1
62.48.128.0	192.168.10.90	255.255.255.224	UG	40	0	0	eth1
192.21.71.0	192.168.10.90	255.255.255.0	UG	40	0	0	eth1
172.25.52.0	172.25.52.240	255.255.255.0	UG	40	0	0	eth1
172.25.52.0	0.0.0.0	255.255.255.0	U	40	0	0	eth1
192.168.10.0	192.168.10.1	255.255.255.0	UG	40	0	0	eth1
192.168.10.0	0.0.0.0	255.255.255.0	U	40	0	0	eth1
195.245.135.0	192.168.10.58	255.255.255.0	UG	40	0	0	eth1
172.27.0.0	192.168.10.90	255.255.0.0	UG	40	0	0	eth1
172.30.0.0	192.168.10.58	255.255.0.0	UG	40	0	0	eth1
172.28.0.0	192.168.10.90	255.255.0.0	UG	40	0	0	eth1
180.142.0.0	180.142.169.196	255.255.0.0	UG	40	0	0	eth0
180.142.0.0	0.0.0.0	255.255.0.0	U	40	0	0	eth0
141.29.0.0	180.142.254.254	255.255.0.0	UG	40	0	0	eth0
10.221.0.0	192.168.10.58	255.255.0.0	UG	40	0	0	eth1

Tabela de *routing* de um *router* Cisco

```
Router> show ip route

Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS interarea
       * - candidate default, U - per-user static route, o - ODR
       P - periodic downloaded static route

Gateway of last resort is not set

R    172.17.0.0/16 [120/1] via 192.168.1.5, 00:00:02, Serial4/1:10.20
S    192.168.55.0/24 [1/0] via 192.168.250.1
     192.168.250.0/30 is subnetted, 1 subnets
C       192.168.250.0 is directly connected, Serial4/0:1.16
     10.0.0.0/32 is subnetted, 2 subnets
R       10.100.100.1 [120/1] via 192.168.1.5, 00:00:02, Serial4/1:10.20
C       10.200.200.1 is directly connected, Loopback1
     192.168.51.0/32 is subnetted, 1 subnets
R       192.168.1.96 [120/1] via 192.168.1.5, 00:00:03, Serial4/1:10.20
C       192.168.1.4 is directly connected, Serial4/1:10.20
R       192.168.1.248 [120/1] via 192.168.1.5, 00:00:03, Serial4/1:10.20
R       192.168.1.252 [120/1] via 192.168.1.5, 00:00:05, Serial4/1:10.20
R       192.168.1.240 [120/1] via 192.168.1.5, 00:00:05, Serial4/1:10.20
R       192.168.1.244 [120/1] via 192.168.1.5, 00:00:05, Serial4/1:10.20
     192.168.2.0/30 is subnetted, 1 subnets
R       192.168.2.0 [120/1] via 192.168.1.5, 00:00:05, Serial4/1:10.20
```

Entrada da tabela de *routing* num *router*

R 10.100.100.1 [120/1] via 192.168.1.5, 00:00:02, Serial4/1:10.20

R – Forma como o caminho foi aprendido (protocolo RIP)

10.100.100.1 – Endereço destino (obs: a máscara está na linha acima, indicada para o grupo de rotas)

[120/1] – Custo [distância administrativa / métrica], para atingir o endereço destino

192.168.1.5 – Endereço para onde enviar o datagrama

00:00:02 – Há quanto tempo foi este caminho aprendido

Serial4/1:10.20 – Interface por onde enviar os datagramas

Distância administrativa (*Administrative Distance*)

- Contexto
  - Um *router* pode aprender rotas por processos diferentes e que usam métricas diferentes.
    - Ex.: Rotas estáticas e vários protocolos de *routing* (RIP, OSPF, BGP)
  - Métricas diferentes não são comparáveis

- Conceito
  - Valor que permite estabelecer preferências entre duas rotas (possivelmente por caminhos diferentes) para o mesmo destino – Rede – aprendidas por processos de *routing* diferentes

Distância administrativa / Preferência de protocolo

	Cisco	Juniper M
Directamente ligada	0	0
IF Local	-	0
Rota estática	1	5
External BGP	20	170
EIGRP	90	-
IGRP	100	-
OSPF	110	
IS-IS	115	15, 18
RIP v1, v2	120	100
Internal BGP	200	170

Tabela de *routing* de um *router* Juniper

```
Router> show ip route
Protocol/Route type codes:
  I1- ISIS level 1, I2- ISIS level2,
  I- route type intra, IA- route type inter, E- route type external,
  i- metric type internal, e- metric type external,
  O- OSPF, E1- external type 1, E2- external type2,
  N1- NSSA external type1, N2- NSSA external type2
   Prefix/Length     Type      Next Hop        Dist/Met        Intf
------------------ ------- -------------- -------------- -------------
10.1.12.0/30       Connect 10.1.12.1       0/1            atm0/0:1.112
10.1.13.0/30       Connect 10.1.13.1       0/1            atm0/0:1.113
10.1.14.0/30       Connect 10.1.14.1       0/1            atm0/0:1.114
10.1.19.0/30       Connect 10.1.19.1       0/1            atm0/0:1.119
10.1.23.0/30       Rip     10.1.12.2       120/2          atm0/0:1.112
                           10.1.13.2       120/2          atm0/0:1.113
10.1.24.0/30       Rip     10.1.12.2       120/2          atm0/0:1.112
10.1.29.0/30       Rip     10.1.12.2       120/2          atm0/0:1.112
                           10.1.19.2       120/2          atm0/0:1.119
10.1.34.0/30       Rip     10.1.13.2       120/2          atm0/0:1.113
10.1.37.0/30       Rip     10.1.13.2       120/2          atm0/0:1.113
10.1.38.0/30       Rip     10.1.13.2       120/2          atm0/0:1.113
33.1.1.1/32        Connect 33.1.1.1        0/1            loopback0
33.1.1.2/32        Rip     10.1.12.2       120/2          atm0/0:1.112
33.1.1.9/32        Rip     10.1.19.2       120/2          atm0/0:1.119
```

Origem da informação de *routing*

- Actualização de caminhos
  - Actualização manual (**rotas estáticas**)
    - Comandos de administração para gerir a tabela de *routing* (redes com poucas mudanças)
  - Actualizações automáticas (**rotas dinâmicas**)
    - Protocolos de troca de informação de encaminhamento entre *routers* (redes com mudanças frequentes)
- Processo de inicialização (depende do Sistema Operativo)
  - Leitura de informação persistente em disco [configuração]
  - Execução de comandos em *scripts* de *boot* [manual]
  - Contactando os *routers* vizinhos [automática]

Gestão de tabelas de *routing* (IOS Cisco)

```
Bogota(config)#   ip route ?
  A.B.C.D  Destination prefix
  profile  Enable IP routing table profile
  vrf      Configure static route for a VPN Routing/Forwarding instance

Bogota(config)#   ip route 20.1.1.0 ?
  A.B.C.D  Destination prefix mask

Bogota(config)#   ip route 20.1.1.0 255.255.255.0 ?
  A.B.C.D        Forwarding router's address
  FastEthernet   FastEthernet IEEE 802.3
  Loopback       Loopback interface
  Null           Null interface
  Serial         Serial

Bogota(config)#   ip route 20.1.1.0 255.255.255.0 10.1.1.254 ?
  <1-255>    Distance metric for this route
  name       Specify name of the next hop
  permanent  permanent route
  tag        Set tag for this route
  <cr>
```

Gestão de tabelas de *routing* (Windows) - 1

```
ROUTE [-f] [-p] [command [destination] [MASK netmask] [gateway] [METRIC metric]]
    [IF interface]

  -f            Clears the routing tables of all gateway entries.  If this is
                used in conjunction with one of the commands, the tables are
                cleared prior to running the command.
  -p            When used with ADD command, makes a route persistent across boots of
                   the system.
  command       Must be one of four:
                  PRINT     Prints   a  route
                  ADD       Adds     a  route
                  DELETE    Deletes  a  route
                  CHANGE    Modifies an existing route
  destination   Specifies the destination host.
  MASK          Specifies that the next parameter is the 'netmask' value.
  netmask       Specifies a subnet mask value to be associated
                with this route entry. If not specified, it defaults to
                255.255.255.255.
  gateway       Specifies gateway.
  METRIC        Specifies that the next parameter 'metric' is the
                cost for this destination
```

Gestão de tabelas de *routing* (Windows) - 2

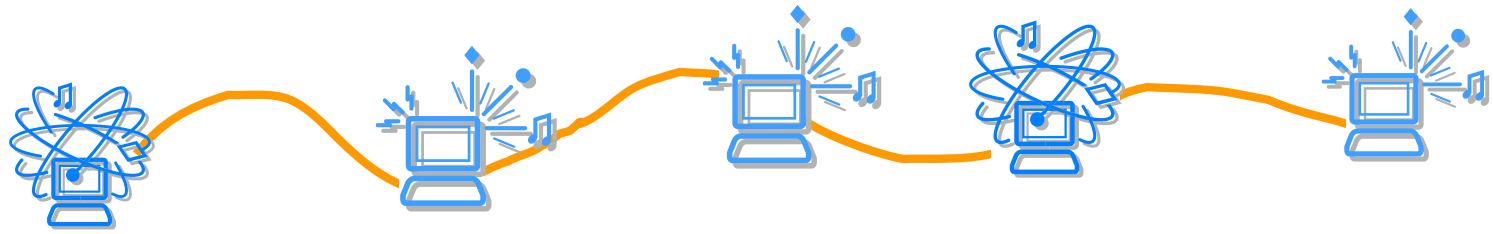
Examples:

```
C:\>route print
Active Routes:
Network Destination        Netmask          Gateway       Interface  Metric
          0.0.0.0          0.0.0.0   141.29.155.254  141.29.155.152       1
     141.29.155.0    255.255.255.0   141.29.155.152  141.29.155.152       1
Persistent Routes:  None

C:\WINNT\system32>route ADD 157.0.0.0 MASK 255.0.0.0 141.29.155.250 METRIC 3
C:\WINNT\system32>route -p ADD 200.0.0.0 MASK 255.0.0.0 141.29.155.245 METRIC 3

C:\>route print
Active Routes:
Network Destination        Netmask          Gateway       Interface  Metric
          0.0.0.0          0.0.0.0   141.29.155.254  141.29.155.152       1
     141.29.155.0    255.255.255.0   141.29.155.152  141.29.155.152       1
        157.0.0.0        255.0.0.0   141.29.155.250  141.29.155.152       3
        200.0.0.0        255.0.0.0   141.29.155.245  141.29.155.152       3
Persistent Routes:
  Network Address          Netmask  Gateway Address  Metric
        200.0.0.0        255.0.0.0   141.29.155.245       3
```

Historial

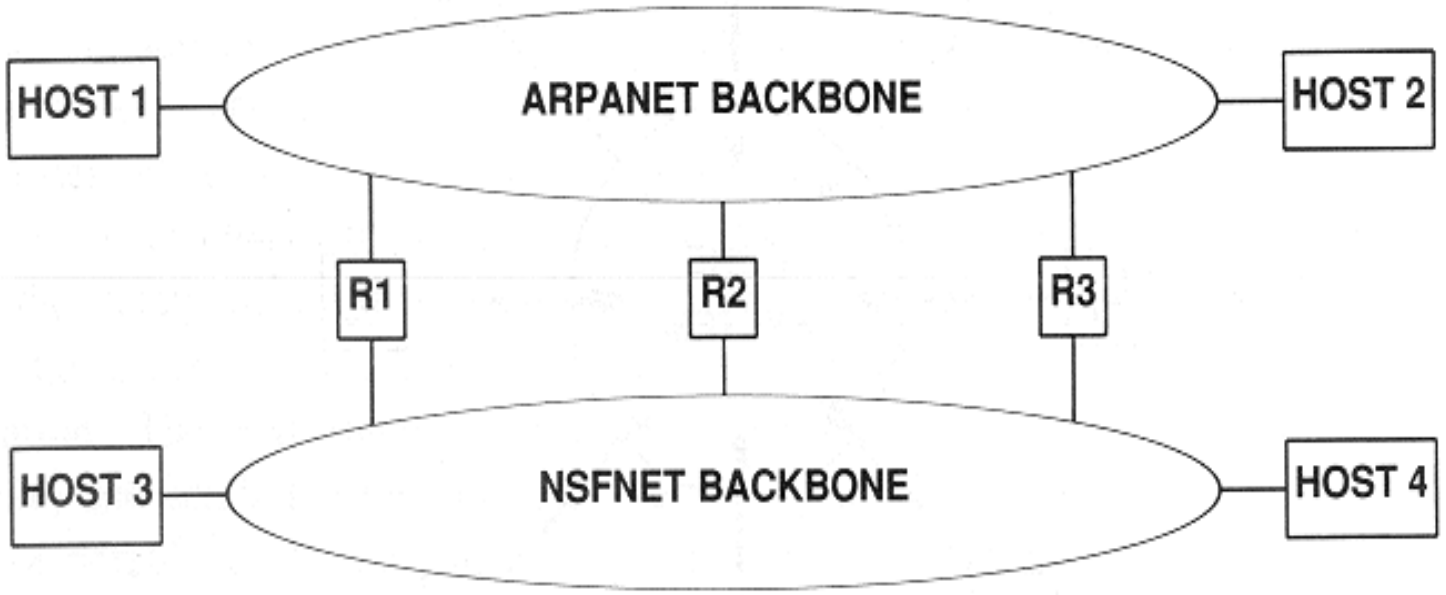


Arquitectura inicial da Internet

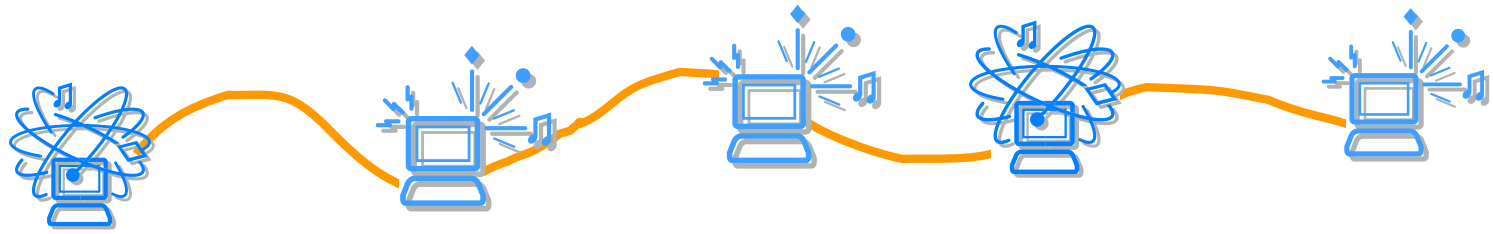
- Todos os *sites* ligavam directamente à ARPANET
- Toda a manutenção das tabelas de *routing* era manual
- Rapidamente os processos tiveram que se tornar automáticos.
- Estrutura inicial
  - Pequeno núcleo de *routers* centrais (*core routers*) com informação total
  - Restantes *routers* continham informação parcial (caminho por omissão)
    - Vantagens
      - Administração local sem afectar a restante Internet
    - Desvantagem
      - Introdução de um possível ponto de inconsistência, podendo tornar certos destinos inatingíveis.

Encaminhamento na Internet inicial (3)

- *Peer Backbones*



Propagação automática de caminhos



Propagação automática de caminhos

- Motivação
- Com o crescimento da Internet foi necessário criar mecanismos (protocolos) de propagação automática de caminhos – ***routing protocols***.

- Facilidades
  - Permitem aos *routers* trocar entre si informação sobre destinos conhecidos
  - Permitem actualizar dinamicamente a informação de encaminhamento quando há alterações na topologia da rede ou em padrões de tráfego

Convergência dos Protocolos de *routing*

- Tempo de Convergência
  - Tempo que leva a todos os *routers*, a executar um determinado protocolo de encaminhamento, a concordarem acerca da topologia da rede depois de uma alteração ter ocorrido (novas rotas ou alteração de rotas existentes)
- Dependências
  - Mecanismo de actualização (*update*)
  - Tamanho da topologia da rede
  - Algoritmo da cálculo de rotas
  - Velocidade dos meios

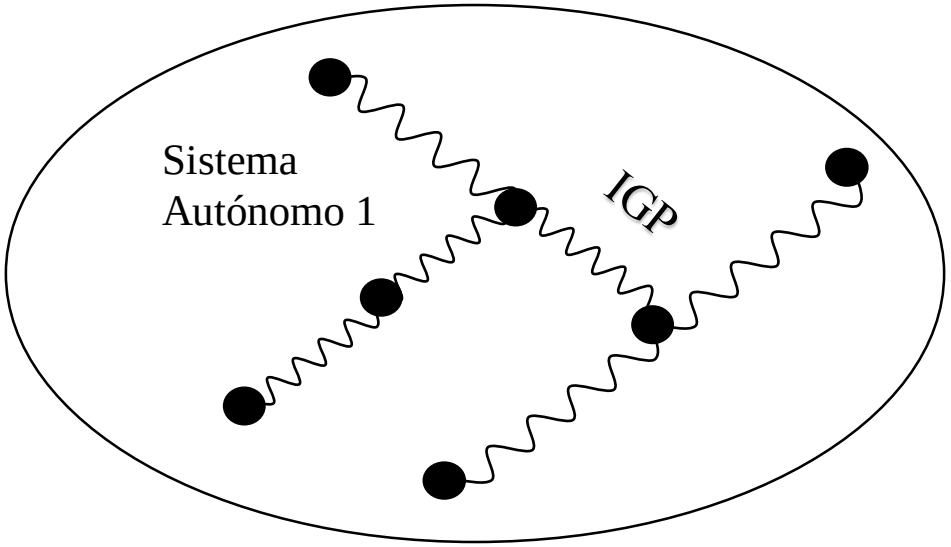
Métricas dos protocolos de *routing*

- Permitem aos *routers* quantificar a qualidade de caminhos alternativos (redundantes) para cada destino e decidir sobre qual deles usar.

- Exemplo de métricas usadas pelos *routers:*
  - Número de *hops*
  - Atraso temporal (*delay*)
  - Custo – Valor administrativo arbitrário
  - Largura de banda – Velocidade de transmissão
  - Carga
  - Fiabilidade
  - MTU – *Maximum Transfer Unit*

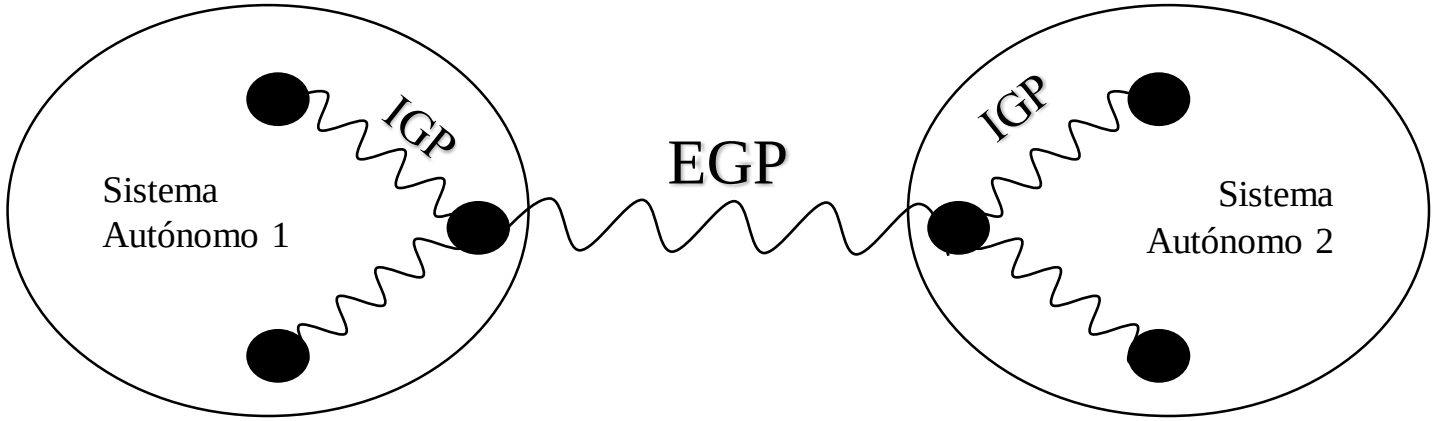
Conceito de Sistema Autónomo

- Sistema autónomo (*Autonomous system* – AS)
  - Conjunto de redes e *routers*, cuja administração do encaminhamento é gerida pela mesma entidade. A cada AS é atribuído um identificador único de 16 bit pela IANA (*Internet Assigned Numbers Authority*)
- Sistema autónomo [protocolo IGP]
  - Conjunto de *routers* a executar um determinado protocolo de IGP



Routers do Sistema Autónomo

- Tipos de *routers* num Sistema autónomo
  - Interiores – ligações para dentro do AS
    - Trocam informação de *routing* referente ao AS.
    - Trocam informação com o *router* exterior do AS a que pertencem
    - Executam protocolos IGP (RIP 2, OSPF, IS-IS, EIGRP)
  - Exteriores – uma ou mais ligações para fora do AS
    - Trocam informação com *routers* de outros sistemas autónomos
    - Isolam o AS do exterior, publicando apenas a informação indispensável
    - Executam protocolos IGP (para dentro do AS) e EGP (BGPv4) (para fora do AS)



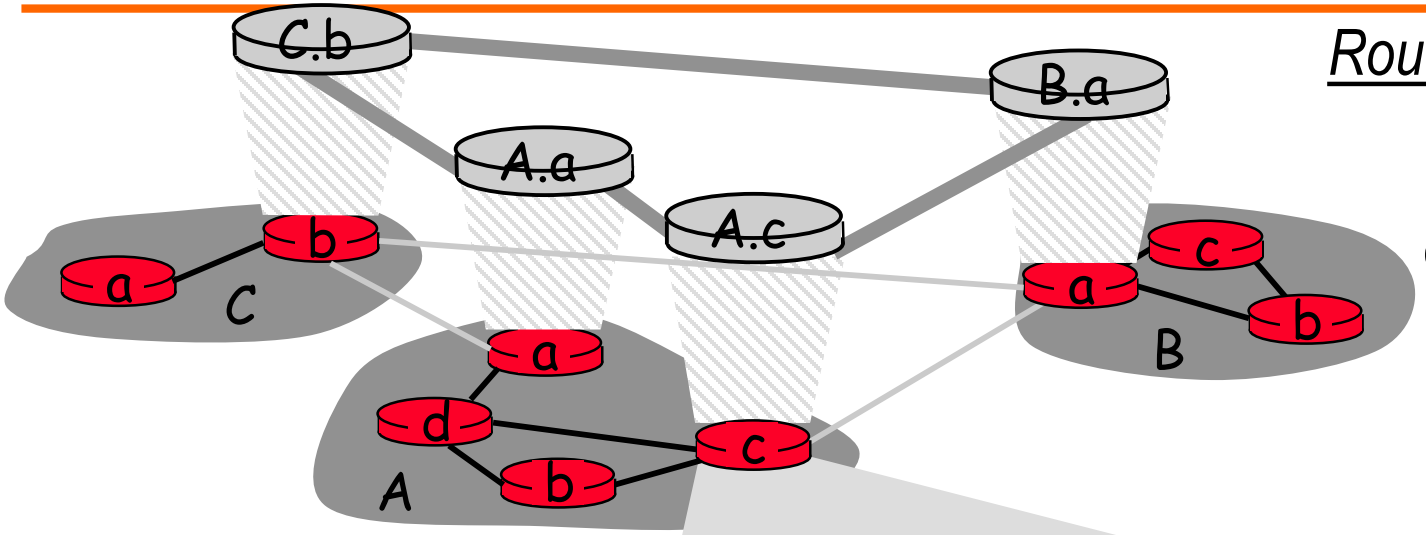
Encaminhamento dinâmico

- A Internet está organizada em sistemas autónomos (*"autonomous systems"* - AS).

- *Interior Gateway Protocols* (IGPs) dentro dos AS.
  - Eg: RIP, OSPF, IS-IS, EIGRP

- *Exterior Gateway Protocols* (EGPs) para encaminhamento de AS para AS.
  - Ex: BGPv4

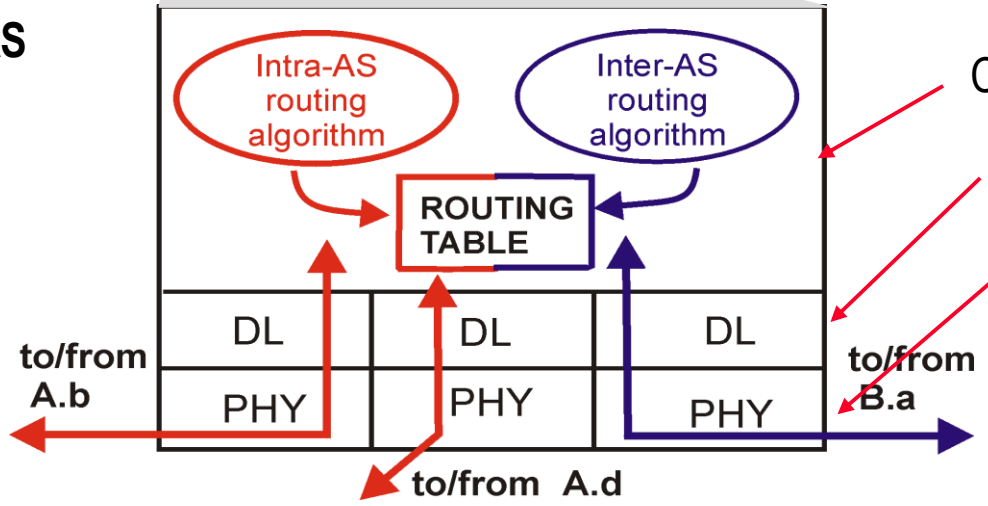
Encaminhamento *Intra-AS* e *Inter-AS*



Routers:

- realizam encaminhamento **inter-AS** com *routers* de outros AS
- realizam encaminhamento **intra-AS** com *routers* do mesmo AS

Encaminhamento **inter-AS** e **intra-AS** no *router* **A.c**

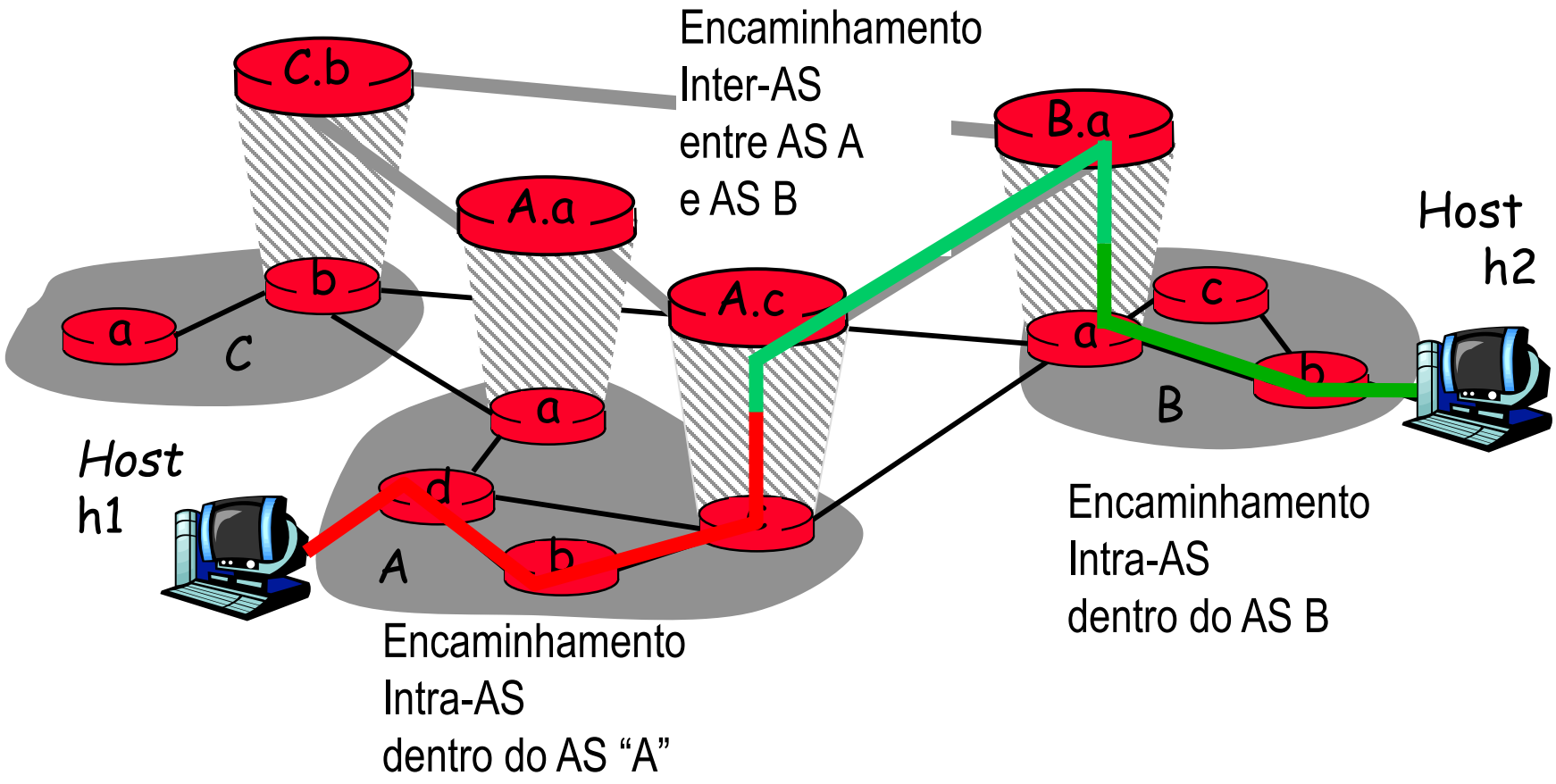


Camada *network*

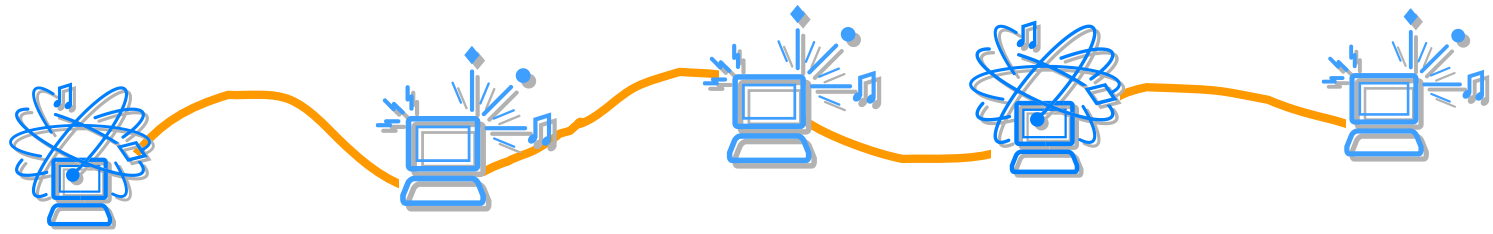
Camada d*ata link*

Camada física

Intra-AS e Inter-AS *routing*: Exemplo



Protocolos de *routing* dinâmico



Distance Vector

Algoritmo Bellman-Ford

***Link State - Shortest Path First* (SPF)**

Algoritmo Dijkstra

Path Vector

Métodos de *routing* dinâmico (*Dynamic Routing Methods*)

- **Definido na origem** (*source-based*): Encaminhamento definido na origem e transportado no datagrama (com o IP, opção *source route*, quantos endereços poderia transportar?).

- ***Link state***: Obter um mapa da rede (em termos de estado das ligações - *link state*) e calcular localmente em cada *router* o melhor caminho para todas as redes (OSPF).

- ***Distance vector***: Cada *router* troca informações com os *routers* vizinhos sobre as redes destino que cada um consegue atingir (endereço de rede e distância) (ex.: RIP).

- ***Path vector***: Encaminhamento "politico" (BGP). Passam a rota completa.

Algoritmos *Vector Distance*

- Os *routers* trocam mensagens entre si do tipo (R, D)
  - (Ex.: “Eu estou a uma distância D da rede R”)

- **Vector** (direcção – R que anuncia) **Distance** (métrica, por ex. nº de saltos)

- Características
  - Os *routers* trocam sempre informação sobre **todos os destinos** que conhecem
  - As mudanças propagam-se de *router* para *router*, podendo entretanto existir *routers* com informação incorrecta
  - Ambientes onde os caminhos mudem rapidamente as tabelas de *routing* podem não estabilizar
    - O algoritmo usado é lento a convergir depois de uma alteração
  - Todos os *routers* têm que entrar no processo

Algoritmos *Vector Distance*

- Vantagens
  - Baixa complexidade do cálculo da tabela de encaminhamento
  - Fácil de implementar
  - Largamente difundido – muitas implementações

- Desvantagens
  - Mensagens de *update* potencialmente muito extensas (Tabela de encaminhamento)
  - As mudanças propagam-se lentamente de *router* para *router*, podendo no entretanto existir *routers* com informação incorrecta
  - O algoritmo usado pode não convergir e é lento quando converge

Encaminhamento *"Distance Vector"* : *Sumário*

- Iterativo, assíncrono: cada iteração local é causada por:
  - Alteração do custo da ligação local
  - Mensagem dum vizinho: um dos seus menores custos alterou-se
- Os vizinhos notificam os seus vizinhos se necessário

Cada nó:

Esperar por (alteração no custo duma ligação local ou mensagem dum vizinho)

Recalcular tabela de distâncias

Se o menor custo dum caminho para qualquer destino se alterar, avisar vizinhos

Algoritmos *Link State* - *Shortest Path First*

- Os *routers* trocam mensagens entre si do tipo (R,X,C)
  - (Ex.: “Conheço uma ligação de R para X com custo C”)
  - **Link** (interface de *router*) **State** (tipo, estado, custos)
- Características
  - Os *routers* trocam informação acerca das **ligações que conhecem**
    - Dados trocados são incrementais – apenas as alterações
      - A dimensão não é proporcional ao nº total de redes.
    - A informação sobre as ligações propaga-se sem alteração.
      - Mais eficiente e mais fácil de detectar falhas
  - Cada *router* tem informação completa acerca da topologia da rede (mapa da área).
    - Cada *router* é um nó de um grafo que representa a rede
  - Cada *router* calcula os caminhos mais curtos independentemente dos outros.
    - Como os cálculos são locais, o algoritmo converge sempre.
    - Uma alteração na topologia obriga a recalcular os caminhos

Protocolos *Link State*

- Objectivo: Criar um “mapa” de toda a rede em cada *router*/nó.

- 1. O nó analisa o estado (*state*) das sua ligações directas e forma um “Link State Packet” (LSP)
- 2. Envia o LSP para todos os nós => chega a todos os outros nós da rede e, depois de todos fazerem o mesmo, cada um dos nós tem um mapa da rede.
- 3. Dado um mapa, corre o algoritmo de Dijkstra (*shortest path algorithm* (SPF)) => obtem os melhores caminhos para todos os destinos
- 4. Tabela de *routing* = próximo salto (*next-hops*) destes caminhos.

Algoritmos *Link State* - *Shortest Path First*

- Vantagens
  - O algoritmo converge rapidamente
  - Imune a *routing* loops
    - Cada *router* tem informação completa acerca da topologia (área)
  - Estrutura hierárquica de *routing* (*backbone* e áreas)
  - Dados trocados são incrementais – apenas as alterações
  - A informação sobre cada ligação é propagada sem alterações
- Desvantagens
  - Utiliza muitos recursos computacionais (CPU, memória)
    - Complexidade elevada do cálculo da tabela de *routing*
    - Várias tabelas: adjacências, topologia, *routing*
  - Complexidade no desenho da topologia da rede (áreas)
  - Descoberta inicial da topologia pode causar tráfego excessivo